[12] The system accurately assesses, controls and optimizes performance of such networks. The invention provides an efficient user interface for installing, configuring and operating various features of the optimization system. Intelligence objects operate at the server node level to dynamically analyze system processes at each server node. The analysis of system processes is extensive and includes hardware, software, operating system and communications.

[13] One feature allows an object to generate a number representing a local utilization value. The local utilization value is a measure of one or more performance factors in the platform hosting the object. The local utilization value can be passed to another platform system hosting a second intelligence object. The second intelligence object can generate its own local utilization value or can combine its local utilization value with the passed value to create a composite utilization value that reflects performance of both platforms. Where different values are from different platforms, the system resolves, adjusts, or normalizes the values to achieve a composite value.

In one embodiment the invention provides a method for monitoring the performance of a digital networked system, wherein the system includes first and second platforms. The method comprising generating a first value indicating a characteristic of operation of the first platform; transferring the first value to the second platform; obtaining a second value indicating a characteristic of operation of the second platform; and combining the first and second values into a composite value by adjusting one of the first or second values to account for a difference in operation characteristics between the first and second platforms.

BRIEF DESCRIPTION OF THE DRAWINGS

[14] Fig. 1 shows a prior art network system;

[15] Fig. 2A shows intelligence objects and performance value passing in the present invention;

[16] Fig. 2B illustrates architectural components of the present invention;

[17] Fig. 2C illustrates a network system with multiple platforms;

[18] Fig. 3A illustrates a user interface display to set up node resource pools;

[19] Fig. 3B illustrates a user interface where a user has added specific nodes;

[20] Fig. 3C illustrates the representation of intelligence objects;

[21] Fig. 3D illustrates further organizing of nodes in NRPs into Functional Resource Pools;

[22] Fig. 3E illustrates establishing connectivity and data flow among NRPs, FRPs and nodes;

[23] Fig. 3F illustrates a connection made between FRP 1 and FRP 2;

[24] Fig. 3G shows a subnetwork;

[25] Fig. 3H illustrates a screen shot of a user interface display to allow a user to set-up a DASPO;

[26] Fig. 4A illustrates the Node Listing console;

[27] Fig. 4B illustrates the Graphic View console;

[28] Fig. 4C illustrates the Monitor console;

[29] Fig. 4D illustrates a series graph of the Monitor Console;

[30] Fig. 4E illustrates a balance graph of the Monitor Console;

[31] Fig. 4F illustrates the History Monitor;

[32] Fig. 5A shows the Redirector Deployment and Installation window;

[33] Fig. 5B illustrates the redirector's Remote Set-Up window;

[34] Fig. 5C shows the File Transfer Settings for a file transfer protocol tab;

[35] Fig. 5D shows a destination folder where redirector files are transferred;

[36] Fig. 5E shows a destination folder specified when using a shared network drive to transfer files;

[37] Fig. 5F shows dialog pertaining to launching a remote set-up using a telnet protocol;

[38] Fig. 5G illustrates a portion of the user interface for preparing a redirector;

[39] Fig. 5H shows an HTTP Redirector Configuration screen;

[40] Fig. 5I shows a Create Connection dialog;

[41] Fig. 5J shows a Load Data Link File dialog;

[42] Fig. 5K shows the Data Link Properties window;

[43] Fig. 5L shows the Confirmation dialog;

[44] Fig. 5M shows the Confirmation dialog with security turned on;

[45] Fig. 5N shows the SLO Deployment and Installation window;

[46] Fig. 5O shows the Remote SLO Set-up window;

[47] Fig. 5P is a first illustration specifying controls and parameters for transfer and remote execution functions;

[48] Fig. 5Q is a second illustration specifying controls and parameters for transfer and remote execution functions;

[49] Fig. 5R is a third illustration specifying controls and parameters for transfer and remote execution functions; and

[50] Fig. 5S is a fourth illustration specifying controls and parameters for transfer and remote execution functions.

DETAILED DESCRIPTION OF THE INVENTION

[51] A preferred embodiment of the present invention is incorporated into products, documentation and other systems and materials created and distributed by MetiLinx, Inc. as a suite of products referred to as "Metilinx iSystem Enterprise" system. The Metilinx system is designed to optimize digital networks, especially networks of many computer servers in large Internet applications such as technical support centers, web page servers, database access, etc.

[52] The system of the present invention uses software mechanisms called "intelligence objects" (IOs) executing on the various servers, computers, or other processing platforms, in a network. The intelligence objects are used to obtain information on the performance of a process or processes, hardware operation, resource usage, or other factors affecting network performance. Values are passed among the intelligence objects so that a composite value that indicates the performance of a greater portion of the network can be derived.

[53] Fig. 2A illustrates intelligence objects and value passing. In Fig. 2A, intelligence objects such as 102 and 104 reside in computer servers. Any number of intelligence objects can reside in a server computer and any number of server computers in the n-tiered system can be equipped with one or more intelligence objects. A first type of intelligence object is a software process called a system level object (SLO) that can monitor and report on one or more aspects of other processes or hardware operating in its host computer server. A second type of intelligence object, called a transaction level object (TLO) is designed to monitor transaction load with respect to its host computer or processes executing within the host computer.

[54] In one embodiment, IO 102 measures a performance characteristic of its host computer and represents the characteristic as a binary value. This value is referred to as the "local" utilization value since it is a measure of only the host computer, or of transaction information relating to the host computer. The local utilization value is passed to IO 104. IO 104 can modify the passed value to include a measurement of its own host computer. The modified value is referred to as a "composite" utilization value. The composite utilization